Optimizing Transliteration for Hindi/Marathi to English Using only Two Weights

M L Dhore[1] *S K Dixit*[2] *R M Dhore*[3]
(1)LINGUISTIC RESEARCH GROUP, VIT, Pune, Maharashtra, India
(2) LINGUISTIC RESEARCH GROUP, WIT, Solapur, Maharashtra, India
(3) LINGUISTIC RESEARCH GROUP, PVG COET, Pune, Maharashtra, India
manikrao.dhore@vit.edu,headextcwitsolapur@gmail.com,ruchidhore93@gmail.com

ABSTRACT

Machine transliteration has received significant research attention in last two decades. It is observed that Hindi to English and Marathi to English named entity machine transliteration is comparably less studied. Currently, research work in this domain is carried out by using grapheme based statistical approaches. But, to achieve better accuracy for the transliteration, an adequate bilingual text corpus is a mandatory requirement for statistical approaches.

This paper focuses on Hindi to English and Marathi to English direct machine transliteration of Indian-origin named entities such as proper names, place names and organization names. Proposed phonetic based statistical approach uses phoneme and named entity length as features for supervised learning and transliterates them in English using full consonant based phonetic scheme without support of corpus. This system takes Indian origin named entities as an input in Hindi and Marathi using Devanagari script and transliterates it into English by using only two weights.

KEYWORDS: Machine Transliteration, Named Entity, Full Consonant, Supervised Learning.

Proceedings of the First International Workshop on Optimization Techniques for Human Language Technology, pages 31–48,
COLING 2012, Mumbai, December 2012.

1. Introduction

Historically, India has been a multilingual country and Indian constitution officially recognizes 22 languages with 11 different scripts and 6000 dialects used in different states spread across the country (Mudur 1999). Hindi is the national language of the India and spoken by more than 500 million Indians. Hindi is the world's fourth most commonly used language after Chinese, English and Spanish. Marathi is one of the widely spoken languages in India especially in the state of Maharashtra. Hindi and Marathi uses the "Devanagari" script for writing and draw their vocabulary mainly from Sanskrit. It is challenging to transliterate out of vocabulary (OOV) words like names and technical terms occurring in the user input across languages with different alphabets and sound inventories. Transliteration is the conversion of a word from one language to another without losing its phonological characteristics. Machine transliteration is usually used to support the machine translation (MT) and cross-language information retrieval (CLIR) to convert the named entities (Padariya 2008).

Existing approaches for Named Entity (hence forth denoted as NE) machine transliterations are linguistic-based and statistical-based (Karimi 2011). The linguistic approach uses hand-crafted rules, based on pattern matching which need a linguistic analysis to formulate rules. Statistical approach tries to generate transliterations using statistical methods based on bilingual text corpora. Transliterations are generated on the basis of statistical models, which are derived from the analysis of bilingual text corpora.

Hindi/Marathi to English direct NE machine transliteration is quite difficult due to the many factors such as difference in writing script, difference in number of characters/alphabets, concept of capitalization of leading characters, phonetic characteristics, relative character length, presence of a number of exonyms, endonyms and historical variants for many place names, number of valid transliterations and availability of the parallel corpus (Saha 2008).

In most of the grapheme based statistical methods parallel corpus is used and trained for the adequate number of entries using one of the learning approaches such as HMM, CRF, SVM etc. As shown below, for the NE / विजयराघवगढ़ (vijayrāghavgarh)/ which is place name, character alignment is obtained using the available tools (like Moses, GIZA++, SRILM,) and then, aligned corpus is trained using one or more statistical probability approaches.

Source Language	Target Language
वि ज य रा घ व ग ढ़	vi ja y rā gha v ga rh

The accuracy of statistical methods depends on how good corpus is prepared and how good learning algorithm is. We have not found any complete named entity bilingual corpus for the Indian languages.

2. Pure and Full Consonant

The basic consonant shape in the Indian script always has the implicit vowel /अ(a)/ and hence there is no explicit matra form for the short vowel 'a'. For example, the NE name /कमल (kamal)/ is linguistically written as कमल = क्+अ+म्+अ+ल्+अ = k+a+m+a+l+a.

However, there are equivalent matras for all the other vowels (ा –ā/A/aa, ि-i, ी-I, ु,-u, ू-U, े-e, ै-ai, ो-o, ौ-au), which get attached to the basic consonant shape whenever the corresponding vowel immediately follows the consonant. The written form of a basic consonant without the implicit 'a' vowel either has an explicit shape or it has the graphical sign '्', known as the Halant (or Virama in Marathi) attached to its basic consonant shape (e.g. क्). This is referred to as the 'Halant form' of the consonant or *pure consonant*. The Halant is the vowel अ (a) omission sign. It serves to cancel the inherent vowel अ of the consonant to which it is applied (Mudur 1999).

When Devanagari vowel phoneme 'a' is added to any generic consonant phoneme then the consonant phoneme is called *full consonant*. It is necessary to have inherent 'a' attached to consonant to add the phone of 'a' vowel to it. If inherent 'a' is not added to the independent consonant phoneme, it becomes very difficult to utter its transliteration in English as well as to obtain its back transliteration in Devanagari from English. For example NE /**कमल** (kəməl)/ will be spelled as /kml/ and would be back transliterated as / **क्म्ल**/ which is tri-conjunct *(*Joshi 2003).

Unicode and ISCII character encoding standards for Indic scripts are based on full form of consonants (Singh 2006). Table 1 shows the pure consonants and full consonants with their English equivalent.

Pure Consonant in Devanagari	English Equivalent	Full Consonant in Devanagari	English Equivalent
क्	k	क् + अ = क	ka
ख्	kh	ख् +अ =ख	kha
ग्	g	ग् + अ = ग	ga

TABLE 1 - Pure and Full Consonant

Following are few examples about how to use the pure and full consonant approach.

Pure Consonant Approach	**Full Consonant Approach**
स् + अ + ई = सई (s + a + i = sai)	स + ई = सई (sa + i = sai)
प् + अ + क् +ई = पकी (p + a + k +i = paki)	प + क + ी = पकी (pa + ka + i)

In proposed approach, the input NE for example / विजयराघवगढ़ /, is segmented into basic syllabic units such as वि ज य रा घ व ग ढ़ and transliterated into English by mapping source language phonetic units into target language phonetic unit using full consonant based phonetic mapping scheme. IAST (International Alphabet of Sanskrit Transliteration) converter can be used to obtain the baseline transliteration. The baseline transliteration for NE / विजयराघवगढ़ / is shown below.

Source Language	Baseline Transliteration
वि ज य रा घ व ग ढ़	vi ja ya rā gha va ga rh

The NE /विजयराघवगड/ is made up of three NEs विजय (Vijay), राघव (Rāghav) and गड (garh) respectively. As it is a multi word NE and consists of three segments, there are three breakpoints (less stressed positions) at /ya/, /va/ and /rh/, hence the inherent 'a' mapped by full consonant mapping should be deleted to get the correct transliteration as /vijayrāghavgarh/.

It is challenging task to find out the number of segments in the given NE, in order to find the boundaries of the segments to remove the inherent 'a' attached due to full consonant mapping. The segments in the NE are obtained by assigning two weights based on the number of diacritics used to form a phonetic unit and the length of the source language NE is used as a feature for supervised learning to obtain the multiple classes for the segmentation. Segmentation is used to identify and delete the schwa (less stressed positions) which is a major issue in direct translation without training the corpus.

3. Related Work

Existing basic models for NE machine transliterations are Grapheme-based and Phoneme-based. The grapheme based model treats transliteration as an orthographic process and tries to map the source language graphemes directly to the target language graphemes. Phoneme-based model considers transliteration as a phonetic process. Under such frameworks, transliteration is treated as a conversion from source grapheme to source phoneme followed by a conversion from source phoneme to target grapheme.

The major contributors in the area of machine transliteration in India are C-DAC (Center for Development of Advanced Computing), NCST (National Center for Software Technology) and Indictrans Team. In the early 1980's, the development of GIST (Graphics and Intelligence - Based Script Technology) was a major breakthrough. C-DAC developed the hardware based solution GIST based on ISCII (Indian Script Code For Information Interchange). The second development was Unicode based encoding standard UTF-8 for Indic script (BIS 1991). The third development (2003) was a phonemic code based scheme for effective processing of Indian languages and used successfully in turnkey jobs such as telephone directory in Hindi, bilingual certificate for Mumbai University, and collector voters' list (Joshi 2003). The applications they have localised are Indian Railways Reservation Charts, Mahanagar Telephone Nigams and Bilingual Telephone Directories.

One of the early works on transliteration is done by Arbabi. They combined neural networks and expert systems for Arabic-English pair using phoneme model (Arbabi 1994). Knight and Graehl developed a five stage statistical model to do back transliteration, that is, to recover the original English name from its transliteration into Japanese Katakana (Knight 1997). Stalls used this for back transliteration from Arabic to English (Stalls 1998). Al-Onaizan and Knight have produced a simpler Arabic-English transliterator and evaluated how well their system can match a source spelling (Al-Onaizan 2002). Their work includes an evaluation of the transliterations in terms of their reasonableness according to human judges. Work in the field of Indian Language CLIR was done by Jaleel and Larkey, which was based on their work in English-Arabic transliteration for CLIR [Jaleel 2003]. Their approach was based on Hidden Markov Model using GIZA++. Phoneme-based models, based on weighted finite state transducers (Knight 1997) and Markov window (Jung 2003) considers transliteration as a phonetic process. OM transliteration scheme provided a script representation which is common for all Indian languages (Ganapathiraju 2005). Punjabi machine transliteration for Punjabi language from Shahmukhi to Gurmukhi used the set of transliteration rules (Malik 2006). Sproat presented a formal computational analysis of Brahmi scripts (Sproat 2002-2004). Kopytonenko focused on computational models that perform grapheme-to-phoneme conversion (Kopytonenko 2006). Ganesh, Harsha, Pingali and

Verma have developed a statistical transliteration technique which is language independent. They selected a statistical model for transliteration which is based on Hidden Markov Model alignment and Conditional Random Fields (Ganesh 2008). Sujan Kumar Saha, Partha Sarathi Ghosh, Sudeshna Sarkar, and Pabitra Mitra have proposed a two-phase transliteration methodology (Saha 2008). The transliteration module uses an intermediate alphabet, which is designed by preserving the phonetic properties. Ekbal, Naskar and Bandyopadhyay made significant attempt to develop transliteration systems for Indian languages to English and especially for Bengali-English transliteration (Ekbal 2007-2010). Manoj K. Chinnakotla, Om P. Damani, and Avijit Satoskar have developed a reasonable transliteration system for resource scared languages by judiciously applying statistical techniques to monolingual resources in conjunction with manually created bilingual rule bases (Chinnakotla 2010). The statistical technique used is the Character Sequence Modeling (CSM), called Language Modelling. They have proved that if the word origin is used for the transliteration, then the system performs better than statistical methods. Jong-Hoon Oh approach is based on two transliteration models (Oh 2009). They used three different machine learning algorithms CRF, MIRA and MEM for building multiple machine transliteration engines. Literature survey shows that the maximum word accuracy achieved is 95.5%, for English to Russian (Martin 2009) using grapheme based statistical approach. In India the maximum word accuracy achieved is 91.59% for Hindi to English (Saha 2008) using phonetic model

4. Approach

The objective of the work is to transliterate named entities from Hindi and Marathi into English. Following Hindi/Marathi language related terminologies are used.

Akshara - It is the minimal articulatory unit of speech in Hindi/Marathi (Pandey 1990).

Swara – It is a pure vowel in Devanagari. Swaras is plural of Swara.

Vyanjana – It is a consonant in Devanagari. Vyanjanas is the plural form of it.

Jodaakshar – It is the conjugates in Devanagari.

Syllable - It is made up of phonetic units to establish minimum rhythm.

Schwa - The schwa is the vowel sound in many lightly pronounced unaccented syllables in words of more than one syllable. It is represented by /ə/ symbol (Naim 2009).

The overall process is carried out as follows.

Step 1: Preparation of phonetic map table for Devanagari to English transliteration using full consonant approach with local language context.

Step 2: Formation of Devanagari Phonetic Units.

Step 3: Generation of Intermediate Phonetic Code (denoted hence forth by IPC) by mapping Devanagari phonetic units to equivalent phonetic unit using phonetic map.

Step 4: Pruning of inherent 'a' generated by vowel matras due to full consonant approach as well as half consonants 'a`' generated by conjuncts (Jodakshar) from intermediate code. (Outcome of this step is denoted hence forth as Modified Intermediate Phonetic Code MIPC).

Step 5: Empirical analysis for multiword named entity formation in Hindi and Marathi.

Step 6: Statistical Model.

Step 7: Segmentation and classification using supervised learning

Step 8: Transliteration Example

4.1 Preparation of phonetic map table

The possibility of different scripts between the source and target languages is the problem that transliteration systems need to tackle. Hindi/Marathi uses the Devanagari script whereas English uses the Roman script. Devanagari script used for Hindi have 12 pure vowels, two additional loan vowels taken from the Sanskrit and one loan vowel from English. According to Cambridge Advanced Learner's Dictionary English has only five pure vowels but, the vowel sound is also associated with the consonants w and y (Koul 2008). There are 34 pure consonants, 5 traditional conjuncts, 7 loan consonants and 2 traditional signs in Devanagari script and each consonant have 14 variations through integration of 14 vowels while in Roman script there are only 21 consonants. The 34 pure consonants and 5 traditional conjuncts along with 14 vowels produce 546 different alphabetical characters (Mudur 1999). Table 2 show 15 vowels along with their matra signs and 34 pure consonants in Devanagari script. The consonant /ळ/ is used only in Marathi language.

Vowel	Matra	Vowel	Matra	Pure consonants				
अ	No matra	ॠ	◌ॄ	क	ख	ग	घ	ङ
आ	◌ा	ए	◌े	च	छ	ज	झ	ञ
इ	◌ि	ऐ	◌ै	ट	ठ	ड	ढ	ण
ई	◌ी	ओ	◌ो	त	थ	द	ध	न
उ	◌ु	औ	◌ौ	प	फ	ब	भ	म
ऊ	◌ू	अं	◌ं	य	र	ल	व	श
ऋ	◌ृ	अ:	◌:	ष	स	ळ	ह	
Loan Vowel		ऑ	◌ॅ					

TABLE 2 - Vowels and Consonant in Hindi and Marathi

Table 3 shows the 5 traditional conjuncts, 7 loan alphabets, 2 traditional signs and 2 special nasal signs in Devanagari script (Walambe 1990].

Traditional Conjuncts	क्ष त्र ज्ञ श्र द्य
Additional Consonants	ड़ ढ़
Loan Consonants	क़ ख़ ग़ ज़ फ़
Traditional Signs	ॐ श्री
Special nasal	◌ं ◌ँ

TABLE 3 - Traditional Conjuncts in Hindi and Marathi

Table 4 shows the phonetic based mapping scheme used to transliterate Devanagari consonant and vowel phones into equivalent English Phones using full consonant approach. It includes all alphabets of Devanagari script used in both Hindi/Marathi and fully based on the National Library of Kolkata and ITRANS of IIT Madras, India (Unicode 2007). It is to note that the first vowel /अ/ in Hindi/Marathi is mapped to English letter ‘a’ (short vowel) while the second vowel /आ/ is mapped to ‘ā’ (long vowel as per IPA) in English. The alphabet ‘a’ in English is a short vowel equivalent to /अ/ which is also a short vowel in Devanagari while /आ/ in Devanagari is a long vowel and mapped capital ‘ā’ or ‘A’ in our phonetic scheme to generate the IPC.

व्यंजनवर्ण CONSONANT PHONEME			Glo- ttal	कोमल तालव्य Velar	कठिन तालव्य Palatal	मूर्धन्य Retroflex	दन्त्य Dental	ओष्ठय Labial
स्पर्श Stops	अघोष Voiceless	अल्पप्राण Unaspirated		क क़ ka qa	च cha	ट Ta	त ta	प pa
		महाप्राण Aspirated		ख ख़ kha, khha	छ Cha	ठ Tha	थ tha	फ फ़ pha fa
	सघोष Voiced	अल्पप्राण Unaspirated		ग ग़ ga, ghha	ज ज़ ja za	ड ड़ Da Dha	द da	ब ba
		महाप्राण Aspirated		घ gha	झ jha	ढ ढ़ Dha , rha	ध dha	भ bha
नासिक्य Nasals				ङ nga	ञ nya	ण Na	न na	म ma
Anuswara and Anunasik				M (default)	◌ं or ◌ँ N (depends on next consonant)			◌ं or ◌ँ
अर्धस्वर वर्ण Semivowels					य ya	र ra	ल la	व va/wa
संघर्षी Fractives	अघोष Voiceless		: -h		श sha	ष Sha	स sa	
	सघोष Voiced		ह ha	जिव्हामूलीय विसर्जनीय				उपध्मानीय
स्वरवर्ण VOWEL PHONEMES	ह्रस्व Short		अ a		इ i	ऋ Ru	ऌ lRu	उ u
	दीर्घ Long		आ A/ ā		ई ए I/ee e	ॠ RU	ॡ lRU	ऊ ओ U oo
संयुक्त Diapthongs					ऐ ai			औ au/ou
पारंपारिक जोडाक्षरे Traditional Conjuncts			क्ष ksha	ज्ञ dnya	द्य dya	श्र त्र shra tra	ॐ om	श्री Shree
स्वतंत्र वर्ण Independent Consonant								ळ La

TABLE 4 - Full Consonant based Phonetic Scheme

4.2 Formation of Devanagari Phonetic Units

As Unicode uses full consonant approach it treats Devanagari consonant phoneme and vowel phoneme as a separate units as shown below.

विजयराघवगढ़(Vijayrāghavgarh) -> व + ि + ज + य + र + ा + घ + व + ग +ढ़

This feature of Unicode is very useful in the creation of Devanagari Phonetic Units. From internal representation of Unicode, phonetic units are formed for Devanagari names as shown below.

विजयराघवगढ़ -> वि | ज | य | रा | घ | व | ग | ढ़

4.3 Generation of Intermediate Phonetic Code

The phonetic scheme is based on full consonant approach and dependent on vowel matra 'अ'. The inherent 'a' is added even if any other vowel matra is present with Devanagari phoneme unit. The script generated in English for Devanagari phonetic unit with inherent 'a' using the phonetic mapping scheme is denoted as Intermediate Phonetic Code (IPC). Table 5 shows how IPC in English is generated for the Devanagari consonant phoneme 'क' when it is combined with different vowel phoneme of Devanagari script.

Devanagari Consonant	Devanagari Vowel	Devanagari Vowel Matra	Devanagari Syllabic Unit	IPC in English
क	अ	No Matra	क	ka
क	आ	ा	का	kaA
क	इ	ि	कि	kai
क	ई	ी	की	kaI
क	उ	ु	कु	kau
क	ऊ	ू	कू	kaU
क	ऋ	ृ	कृ	kaRu

TABLE 5 -IPC for Devanagari Consonant 'क'

The following method is used to generate the IPC in English.

- Devanagari name divided into the syllabic units are called as Source Transliteration Units (STU). STU is equivalent to phonetic unit of Devanagari. The STU can be represented using following regular expression.
  STU = ((V) | (C) | (CV) | (CCV) | (C...CV)) (G)
  where C = Consonant, V = Vowel and G = Nasalization of vowels
- English name divided into the syllabic units called as Target Transliteration Units (TTU). TTU is equivalent to a phonetic unit of English. The regular expression for English can be written as TTU = C*V*
- The Devanagari name is represented as a collection of Devanagari phonetic units.
  Name in Devanagari = { STU_1, STU_2, ... STU_n }
- The English name is represented as a collection of English phonetic units.
  Name in English = { TTU_1, TTU_2, ... TTU_m }
- IPC is obtained by using direct mapping of STUs to TTUs on one to one basis.

Table 6 shows the few examples for IPC in English for the Devanagari NE.

NE	STUs	TTUs	IPC in English
नोवरोझाबाद	नो|व|रो|झा|बा|द	nao|va|rao|jhaā|baā|da	naovaraojhaābaāda

TABLE 6 - IPC Examples

4.4 Pruning

An IPC is generated by mapping STUs to TTUs. STUs are generated from the Devanagari name which uses Unicode encoding. Due to the full consonant nature of Unicode, there is an inherent ‘a’ followed by consonant phoneme for all Devanagari vowel matras in IPC form. The inherent ‘a’ generated for Devanagari phonetic units having any vowel matra should be removed. One of the problems in Devanagari to English transliteration is the transformation of conjugates. When the half consonant cluster (Virama/Halant ्) appears in Devanagari script, it is mapped in English with the letter symbol ‘a`’ which appears in between two consonant phonemes. There is no practice to represent such half consonant in English. All the viramas in Devanagari script mapped as a half consonants ‘a`’ should be eliminated from the IPC. The output after pruning inherent ‘a’ and ‘a`’ is denoted as Modified IPC (MIPC). Table 7 shows the example of removal of inherent ‘a’ if matra or half consonant is present.

STU	TTU	If matra , remove ‘a’	If conjunct, remove ‘a`	Modified IPC
कै	kaai	kai	---	kailāshanātha
ला	laā	lā	---	
श	sha	---	---	
ना	naā	nā	---	
थ	tha	----	---	

TABLE 7 - IPC to MIPC for Devanagari NE ‘Kailashnath’

4.5 Empirical analysis for multiword named entity formation

An example shown in Table -7, the NE /*कैलाशनाथ*/ is transliterated as /*kailāshanātha*/. The NE /*कैलाशनाथ*/ is multi-word name consists of /*कैलाश*/ and a suffix /*नाथ*/ . An ‘a’ followed by ‘sh’ should be deleted as well as the last ‘a’ also should be deleted to obtain the correct transliteration.

कैलाशनाथ → [कै | ला | श | ना | थ] → [kaai | laā| sha | naā | tha] → [kai | lā | sha | nā | tha] → [kai | lā| sh | nā | th] →[kailāsh | nāth] →kailāshnāth

It has been observed that the minimum length of the NE is 1 akshara (formed using 1 syllabic unit) and maximum length is 8 aksharas. There are very few named entities consisting one syllable. From the number of aksharas in the named entities, 8 categories are made. One akshara is considered equivalent to one phonetic unit in the Devanagari NEs. It is found that nearly 50% NEs used in India are a combination of two or more individual named entities (denoted hence forth NEs). For one akshara, two aksharas and three aksharas NEs, transliteration is quite simple. As the length of a NE increases, the segmentation becomes important to find out the number of words used to form the NE in order to separate the rhythms within it and in turn number of phonetic units in each rhythm.

Most of the four aksharas, five aksharas, six aksharas, seven aksharas and eight aksharas NEs are formed with the combination of two or three different rhythmic units. Table 8 shows the observations of possible combinations of phonetic segments from pronunciation point of view.

NE	Segmentation	Segment Lengths
Number of Aksharas =4		
श्रीवर्धन(Shriwardhan)	श्री + वर्धन(Shrī + wardhan)	1 + 3
बाजीराव(Bājīrāo)	बाजी + राव (Bājī+ rāo)	2 + 2
धवलश्री(Dhawalshrī)	धवल + श्री (Dhawal + shrī)	3 + 1
Number of Aksharas =5		
मनमोहन(Manmohan)	मन + मोहन (Man + mohan)	2 + 3
माणिकराव(Mānikrāo)	माणिक + राव (Mānik + rāo)	3 + 2
श्रीनारायण(Shrinārāyan)	श्री + नारायण (Shrī + nārāyan)	1 + 4
Number of Aksharas =6		
करमरकर(Karmarkar)	कर+मर+ कर (Kar + mar + kar)	2 + 2 + 2
प्रेमनारायण(Premnārāyan)	प्रेम + नारायण (Prem + nārāyan)	2 + 4
भारतभुषण(Bhāratbhushan)	भारत+ भुषण (Bhārat + bhushan)	3 + 3
जनार्दनराव(Janārdhanrāo)	जनार्दन + राव (Janārdhan + rāo)	4 + 2
Number of Aksharas =7		
राजगुरुनगर(Rajgurunagar)	राज+गुरु+नगर(Raj+guru+nagar)	2 + 2 + 3
मनमाधवराव(Manmādhavrāo)	मन+माधव+राव(Man+mādhav+rāo)	2 + 3 + 2
पंढरपुरकर(Pandharpurkar)	पंढर + पुर + कर(Pandharpurkar)	3 + 2 + 2
प्रकाशनारायण(Prakāshnārāyan)	प्रकाश+नारायण(Prakāsh + nārāyan)	3 + 4
गिरिराजकिशोर(Girirājkishor)	गिरिराज + किशोर(Girirāj + kishor)	4 + 3
पुरुषोत्तमदास(Purushottamdās)	पुरुषोत्तम+ दास(Purushottam + dās)	5 + 2
Number of Aksharas =8		
विजयराघवगढ़(Vijayrāghavgarh)	विजय+राघव+गढ़ (Vijay+rāghav+garh)	3 +3 +2
नारायणगावकर(Nārāyangāvkar)	नारायण+गाव+कर(Nārāyan +gāv+kar)	4 +2+ 2
पुरुषोत्तमनगर (Purushottamnagar)	पुरुषोत्तम+नगर (Purushottam+nagar)	5 + 3
त्रिभुवननारायण (Tribhuvannārāyan)	त्रिभुवन+ नारायण (Tribhuvan+nārāyan)	4 + 4

TABLE 8 –Segment Length analysis of Multiword Named Entities

Empirical analysis given in Table 8, confirms that there are always minimum two segments in four to eight aksharas NE. These observations are useful to find out the stressed and unstressed syllables in the multi word NEs which is required to find the break points. These break points are referred to as schwa positions, where the occurrence of 'a' vowel is not pronounced and hence deleted.

4.6 Statistical Model

Following terminologies are used in the statistical model

Input: NE in source language is denoted by source word (SW)

Output: NE in the target language as output (English) is denoted by target word (TW).

- SW is divided into the phonetic units called Source Transliteration Units (STUs).
- The TTUs is used to denote Target Transliteration Units (TTU). TTU is equivalent to phonetic unit of English after mapping STU.

- The Devanagari name is represented as a collection of Devanagari phonetic units.
- Name in Devanagari = STU_1, STU_2, ... STU_n }
- The English name is represented as a collection of English phonetic units.
- Name in English = { TTU_1, TTU_2, ... TTU_m }
- WSTU is the weight assigned to a phonetic unit (STU) of the source language.
- WTTU is the weight assigned to a phonetic unit (TTU) of the target language.

If the STU contains any of the vowel matra from [◌ा, ि◌ ,◌ी, ◌ु ,◌ू ,◌े ,◌ै,◌ो,◌ौ, ◌ं ◌ः] or complete vowel [अ, आ, इ,ई, उ, ऊ, ए,ऐ,ओ,औ] , weight 2 is assigned to it, otherwise weight 1 is assigned. The same weights are mapped to the corresponding TTUs. Weight assignment is represented using mathematical equations (1) and (2).

$$\boldsymbol{WSTU_i} = \begin{cases} \boldsymbol{2\ if\ \exists STU_i\ with\ vowel\ matra\ or\ complete\ vowel} \\ \boldsymbol{1\ \ if\ \exists STU_i\ without\ vowel\ matra} \end{cases} \qquad (1)$$

$$\boldsymbol{WTTU_i} = \begin{cases} \boldsymbol{2\ if\ \exists TTU_i\ with\ [\mathbf{A, e, i, I, o, u, U, ai, au}]\ \mathbf{or\ preceded\ by\ consonant}} \\ \boldsymbol{1\ \ if\ \exists TTU_i\ with\ \ \mathbf{consonant\ \ followed\ by\ \ short\ vowel}\ [\mathbf{a}]} \end{cases} \qquad (2)$$

where i = 1..n

The probability is calculated for individual phonetic units. As the basis of the method is phonetic model, the probability of mapping STU to TTU is always 1 for the diacritic marks [◌ा, ि◌ ,◌ी, ◌ु ,◌ू ,◌े ,◌ै,◌ो,◌ौ, ◌ं ◌ः] and complete vowels [अ, आ, इ,ई, उ, ऊ, ए,ऐ,ओ,औ] . When a NE written in Devanagari script is transliterated using English script, the implicit अ attached to the single consonant either gets mapped to ‘a’ or null depending on the patterns of stress and intonation in a language. The initial probability for all inherent short vowel /a/ is taken as 0.

$$\boldsymbol{P(STU_i|TTU_i)} = \begin{cases} \boldsymbol{1\ if\ \exists TTU_i : TTU_i = WTTU_i = 2} \\ \boldsymbol{0\ if\ \exists TTU_i : TTU_i = WTTU_i = 1} \end{cases} \qquad \text{where i = 1..n} \qquad (3)$$

If the result of equation (3) is zero for any TTU, then segments are formed (for the NE having aksharas more than 3) if any, and the probability is recalculated for each segment using the TTU position in the word.

$$\boldsymbol{P(STU_i|TTU_i|WORD\_INITIAL_i)} = \begin{cases} \boldsymbol{1\ if\ \exists TTU_i : TTU_i\ \ \exists 'a'} \\ \boldsymbol{0\ \ if\ \exists TTU_i : TTU_i \nexists\ 'a'} \end{cases} \qquad \text{where i=1} \qquad (4)$$

Ex-Or

$$\boldsymbol{P(STU_i|TTU_i|WORD\_MEDIAL_i)} = \begin{cases} \boldsymbol{1\ if\ \exists TTU_i : TTU_i\ \ \exists 'a'} \\ \boldsymbol{0\ \ if\ \exists TTU_i : TTU_i\ \nexists\ 'a'} \end{cases} \qquad \text{where i=2 to n-1} \qquad (5)$$

Ex- Or

$$\boldsymbol{P(STU_i|TTU_i|WORD\_FINAL_i)} = \begin{cases} \boldsymbol{1\ if\ \exists TTU_i : TTU_i \nexists\ 'a'} \\ \boldsymbol{0\ \ if\ \exists TTU_i : TTU_i \exists\ 'a'} \end{cases} \qquad \text{where i= n} \qquad (6)$$

4.7 Segmentation and classification

In our approach, classification is obtained by using the supervised learning approach. Following is the analysis of the named entities in Hindi and Marathi to obtain the classification based on the position and weight of the phonetic entity. Most of the four aksharas named entities are formed with the combination of two different words. As the length of the Devanagari word is 4 and different weights are 2, the possible combinations can be sixteen as shown in Table 9.

SN	Weight Pattern	Segmentation	Named Entity	Segments
1	1111	11 + 11 (2 + 2)	दशरथ (Dashrath)	दश + रथ
2	1121	11 + 21 (2 + 2)	सलमान(Salmān)	सल + मान
3	2111	21 + 11 (2 + 2)	गिरधर(Girdhar)	गिर + धर
4	2121	21 + 21 (2 + 2)	शामराव(Shāmrāo)	शाम + राव
5	1112	11 + 12 (2 + 2)	मनकर्णा(Mankarna)	मन + कर्णा
6	2112	21 + 12 (2 + 2)	शिवदत्त(Shivdatta)	शिव + दत्त
7	1122	11 + 22 (2 + 2)	मनवेंद्र(Manvendra)	मन + वेंद्र
8	2122	21 + 22 (2 + 2)	धृतराष्ट्र(Dhrutrāshtra)	धृत + राष्ट्र
9	1211	12 + 11 (2 + 2)	वरेकर(Varekar)	वरे + कर
10	1221	12 + 21 (2 + 2)	फत्तेलाल(Fattelāl)	फत्ते + लाल
11	2211	22 + 11 (2 + 2)	निलोफर (Nilofar)	निलो + फर
12	1222	12 + 22 (2 + 2)	झकारीया(Zakārīyā)	झका + रीया
13	2221	22 + 21 (2 + 2)	फैजाबाद(Faizābād)	फैजा + बाद
14	1212	12 + 12 (2 + 2)	मनोरमा(Manoramā)	मनो + रमा
15	2212	22 + 12 (2 + 2)	विश्वकर्मा (Vishwakarma)	विश्व +कर्मा
16	2222	22 + 22 (2 + 2)	चंद्रमौली (Chandramaulī)	चंद्र + मौली

TABLE 9 - Weight Patterns for NE of length 4

From Table 9 following two observations, two inferences and two classes are obtained.

Observations from named entities 1 to 8 in Table 9 are

- Weight patterns 1 to 8 have weight 1 at second position.
- The second position has low weight hence the right boundary of the first segment.

Observations from named entities 9 to 16 in Table 9 are

- Weight patterns 9 to 16 have weight 2 at the second position.
- The second position has high weight in the basic pattern.

Inference 1: From a transliteration point of view, for the four aksharas NE, if the weight pattern has weight 1 at second position; it indicates that the NE consists of two segments. In this case, the NE can be divided into two segments of 2 and 2 aksharas, respectively and short vowel 'a' of second akshara which is schwa gets removed.

Inference 2: From a transliteration point of view, for the four aksharas NE, if the weight pattern has weight 2 at the second position, then no segmentation is needed due to high weight.

Classification: *Class I*: Named entities having weight 1 at the second position

Class II: Named entities having weight 2 at the second position

One of the inferences for eight aksharas NE having 3 segments like /विजयराघवगढ़/ is

Inference 3: From a transliteration point of view, for the eight aksharas named entity, if the weight pattern has weights 1211 or 1221 from the position third to the sixth, it indicates that the named entity consists of three segments. In this case, the named entity can be divided into three segments of 3, 3 and 2 aksharas respectively.

Similar analysis is performed for NEs where numbers of aksharas were equal to 5,6,7,8. Summary of the analysis is given in Table 10.

Number of Aksharas	Possible combination	Number of patterns observed	Number of Inferences	Number of classes found
4	16	16	2	2
5	32	24	3	3
6	64	39	4	4
7	128	17	7	7
8	256	8	4	4

TABLE 10 -Summary of Classification

4.8 Transliteration Example

From the statistical analysis of 15224 named entities (Main Data Sources: Voters' list of State of Maharashtra and Atlas of India both of which are available in Marathi/Hindi and English), twenty inferences are drawn. With the help of supervised learning, twenty classes are used for segmentation and finding the schwa identification and deletion. The overall implementation is illustrated by taking the NE of eight aksharas.
Input in Devanagari → विजयराघवगढ़
Phonetic Units in Devanagari→ [वि] [ज] [य] [रा] [घ] [व] [ग] [ढ़]
Phonetic Units in English →[vi][ja][ya][rā][gha][va][ga[][rh]
Weight Assignment (Eq 2) → [2][1][1][2][1][1][1][1]
Initial Probabilities (Eq 3)→[1][0][0][1][0][0][0][0]
Segmentation (Inference 3)→[vi][ja][ya], [rā][gha][va] and [ga][rh] Three Segments
Probabilities (Eq 4,5 and 6) →[1][1][0] , [1][1][0] and [1][0]
Schwa Deletion → Segment ending Schwas
Final Probabilities (Eq 6) → [1][1][1][1][1][1][1][1]
Transliteration in English→ vijayrāghavgarh

5. Experimentation and Results

Table 11 shows the results of the 15224 names transliterated by using phonetic model and statistical approach for segmentation and schwa deletion.

Number of Akshras	Number of Names	Number of Correct Transliteration (Top-1)	Number of Incorrect Transliterations
2	1839	1832 (99.619%)	07 (0.381%)
3	6061	6040 (99.635%)	21 (0.365%)
4	4780	4646 (97.196%)	134 (2.804%)
5	1970	1785 (90.609%)	185 (9.391%)
6	497	442 (88.933%)	55 (11.067%)
7	61	57 (93.442%)	4 (6.558%)
8	16	12 (75%)	4 (25%)
	15224	14814 (97.306%)	410 (2.694%)

TABLE 11 - Results of phonetic model using statistical approach

The results of accuracy (Top-1) for phonetic model using statistical approach are depicted in figure 1.

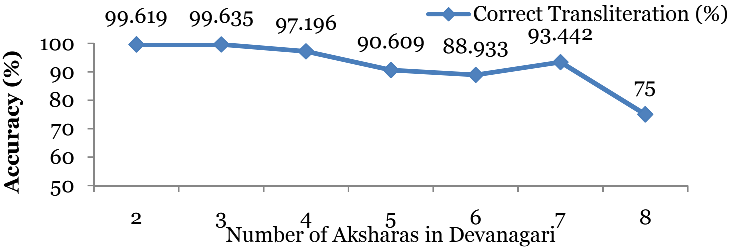

FIGURE 1 - Top-1 Accuracy

Figure 2 depicts the results of Top-1 accuracy, Mean F-Score, Top-2 MRR, Precision and Recall for phonetic model using statistical approach (Li 2009).

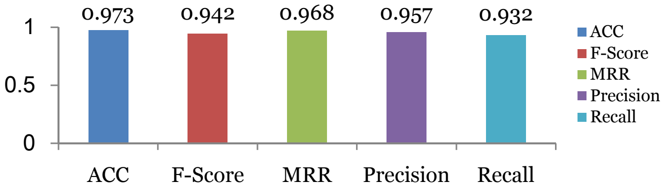

FIGURE 2 - Results Using Performance Metrics

Table 12 is used to generate multiple transliteration candidates (Chinnakotla 2010) .

Hindi	क	ख	ग	ड	व	ई	श	क्ष	औ
English	k,c,q	k,kh	g, gh	d,dh	w,o,b,bh	i,e,ee,ey	sh,s	ksh,x	au,ou

Table 12 : Mapping Table for Multiple Candidates Generation

Conclusion

We presented optimized direct machine transliteration for Hindi to English and Marathi to English language pairs using full consonant approach using only two weights and without corpus. As Hindi and Marathi languages are phonetically rich languages, phonetic based model is used. The accuracy of the transliteration decreases as the length of the Hindi and Marathi named entity increases in terms of number of aksharas. The accuracy of the transliteration decreases if the named entity is made up of multiple smaller length named entities. We showed that, a transliteration system can be built from phonetics, based on local linguistic word formation logic and supervised learning and its accuracy is 97.3%. The phonetic based statistical approach shows the significant improvement in the accuracy for the named entities consisting of four aksharas, five aksharas, six aksharas, seven aksharas and eight aksharas.

References

Al-Onaizan Y, Knight K (2002), Machine translation of names in Arabic text, *Proceedings of the ACL conference workshop on computational approaches to Semitic languages*.

Arbabi M, Fischthal S M, Cheng V C and Bart E (1994), Algorithms for Arabic name transliteration, *IBM Journal of Research and Development*, pp. 183-194.

BIS (1991), Indian standard code for information interchange (ISCII), *Bureau Of Indian Standards*, New Delhi.

Chinnakotla Manoj K., Damani Om P., and Satoskar Avijit (2010), Transliteration for Resource-Scarce Languages, *ACM Trans. Asian Lang. Inform,*Article 14, pp 1-30.

Ekbal A. and Bandyopadhyay S. (2007), A Hidden Markov Model based named entity recognition system: Bengali and Hindi as case studies, *Proceedings of 2nd International conference in Pattern Recognition and Machine Intelligence*, Kolkata, India, pp. 545–552.

Ekbal A. and Bandyopadhyay S. (2008), Bengali named entity recognition using support vector machine, *In Proceedings of the IJCNLP-08 Workshop on NER for South and South East Asian languages*, Hyderabad, India, pp. 51–58.

Ekbal A. and Bandyopadhyay S. (2008), Development of Bengali named entity tagged corpus and its use in NER system, *In Proceedings of the 6th Workshop on Asian Language Resources*.

Ekbal A. and Bandyopadhyay S. (2008), A web-based Bengali news corpus for named entity recognition, *Language Resources & Evaluation*, vol. 42, pp. 173–182.

Ekbal A. and Bandyopadhyay S.(2008), Improving the performance of a NER system by post-processing and voting, *In Proceedings of Joint IAPR International Workshop on Structural Syntactic and Statistical Pattern Recognition*, Orlando, Florida, pp. 831–841.

Ekbal A. and Bandyopadhyay S.(2009), Bengali Named Entity Recognition using Classifier Combination, *In Proceedings of Seventh International Conference on Advances in Pattern Recognition*, pp. 259–262.

Ekbal A. and Bandyopadhyay S. (2009), Voted NER system using appropriate unlabelled data, In *Proceedings of the Named Entities Workshop*, ACL-IJCNLP.

Ekbal A. and Bandyopadhyay S. (2010), Named entity recognition using appropriate unlabeled data, post-processing and voting*, In Informatica*, Volume (34), No. 1, pp. 55-76.

Ganapathiraju M., Balakrishnan M., Balakrishnan N., Reddy R. (2005), .OM: One Tool for Many (Indian) Languages. ICUDL: *International Conference on Universal Digital Library*, Hangzhou.

Ganesh S, Harsha S, Pingali P, and Verma V (2008), Statistical transliteration for cross language information retrieval using HMM alignment and CRF, *In Proceedings of the Workshop on CLIA*, Addressing the Needs of Multilingual Societies.

Jaleel Nasreen Abdul and Larkey Leah S. (2003), Statistical transliteration for English-Arabic cross language information retrieval, *In Proceedings of the 12th international conference on information and knowledge management,* pp: 139 – 146.

Joshi R K, Shroff Keyur and Mudur S P (2003), A Phonemic code based scheme for effective processing of Indian languages, National Centre for Software Technology, Mumbai, *23rd Internationalization and Unicode Conference*, Prague, Czech Republic, pp. 1-17.

Jung S. Y., Hong S., S., Paek E.(2003), English to Korean transliteration model of extended Markov window, *In Proceedings of the 18th Conference on Computational Linguistics*, pp.383–389.

Karimi S, Scholer F, and Turpin(2011), Machine transliteration survey, *ACM Computing Surveys*, Vol. 43, No. 3, Article 17, pp.1-46.

Knight Kevin and Graehl Jonathan (1997), Machine transliteration, *In proceedings of the 35th annual meetings of the Association for Computational Linguistics*, pp. 128-135.

Kopytonenko M. , Lyytinen K. , and Krkkinen T.(2006), Comparison of phonological representations for the grapheme-to-phoneme mapping, In *Constraints* on Spelling Changes: *Fifth International Workshop on Writing Systems*, Nijmegen, The Netherlands.

Koul Omkar N. (2008), Modern Hindi Grammar, Dunwoody Press

Li Haizhou, Kumaran A, Vladimir Pervouchine and Min Zhang (2009), *Report of NEWS 2009 Machine Transliteration Shared Task, ACL-IJCNLP*, pp. 1-19

Malik M.G.A. (2006), Punjabi Machine Transliteration, *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the ACL*, pages 1137–1144.

Martin Jansche and Richard Sproat (2009), Named Entity Transcription with Pair n-Gram Models, *Machine Transliteration Shared Task, ACL-IJCNLP*, pp. 32-35

Mudur S. P., Nayak N., Shanbhag S., and Joshi R. K. (1999), An architecture for the shaping of indic texts, *Computers and Graphics*, vol. 23, pp. 7–24.

Naim R Tyson and Ila Nagar (2009), Prosodic rules for schwa-deletion in Hindi Text-to-Speech synthesis, *International Journal of Speech Technology*, pp. 15–25

Oh Jong-Hoon, Kiyotaka Uchimoto, and Kentaro Torisawa (2009), Machine transliteration using target-language grapheme and phoneme: Multi-engine transliteration approach, Proceedings of the Named Entities Workshop ACL-IJCNLP Suntec, Singapore,AFNLP, pp. 36–39

Padariya Nilesh, Chinnakotla Manoj, Nagesh Ajay, Damani Om P.(2008), Evaluation of Hindi to English, Marathi to English and English to Hindi, *IIT Mumbai CLIR at FIRE.*

Pandey Pramod Kumar (1990), Hindi schwa deletion, Department of Linguistics. South Gujarat University, Surat , lndia, *Lingua* 82, pp. 277-31

Saha Sujan Kumar, Ghosh P. S, Sarkar Sudeshna and Mitra Pabitra (2008), Named entity recognition in Hindi using maximum entropy and transliteration.

Singh Anil Kumar (2006), A computational phonetic model for Indian language scripts, Language Technologies Research Centre International Institute of Information Technology Hyderabad, India.

Sproat R.(2002), Brahmi scripts, In Constraints on Spelling Changes: *Fifth International Workshop on Writing Systems*, Nijmegen, The Netherlands.

Sproat R.(2003), A formal computational analysis of Indic scripts, *In International Symposium on Indic Scripts: Past and Future*, Tokyo.

Sproat R.(2004), A computational theory of writing systems, In Constraints on Spelling Changes: *Fifth International Workshop on Writing Systems*, Nijmegen, The Netherlands.

Stalls Bonnie Glover and Kevin Knight (1998), Translating names and technical terms in Arabic text.

Unicode Standard 5.0 (2007) – Electronic edition, 1991–2007 Unicode, Inc. *Unicode Consortiums*, http://www.unicode.org.

Walambe M. R. (1990), Marathi Shuddalekhan, *Nitin Prakashan*, Pune

Walambe M. R. (1990), Marathi Vyakran, *Nitin Prakashan*, Pune